

CONTROL INTERFACE PROTOCOL FOR TELEPHONE SETS FOR A SATELLITE TELEPHONE SYSTEM

CROSS-REFERENCE TO RELATED APPLICATIONS

Claim of Priority under 35 U.S.C. §120

[0001] The present Application for Patent is a continuation and claims priority to Patent Application No. 09/732,805 entitled "Control Interface Protocol for Telephone Sets for a Satellite Telephone System" filed December 7, 2000, pending, which is a divisional of Patent Application No. 09/201,701 entitled "Control Interface Protocol for Telephone Sets for a Satellite Telephone System" filed November 30, 1998, pending and assigned to the assignee hereof and hereby expressly incorporated by reference herein.

BACKGROUND OF THE INVENTION

I. Field of the Invention

[0002] The present invention relates generally to multi-access communication systems. More specifically, the present invention relates to a control interface protocol for telephone sets for use in a satellite telephone system.

II. Related Art

[0003] Many communication systems, such as satellite systems, multidrop telephone lines, and multitap bus systems use multi-access communication configurations. Generally, in a multi-access communication system, a plurality of nodes are connected to a single multi-access channel or an interface bus. Each node has a queue of packets, representing data to be transmitted over or by the interface bus. The interface bus views all waiting packets as one combined queue to be served by an appropriate protocol.

[0004] Several methods for access to an interface bus have been developed for multi-access communication systems. One method is the "free-for-all" method in which nodes normally send new packets immediately, hoping for no interference from other nodes. The problem, however, is that packets need to be retransmitted when collisions, that is, interference, occur. The other method is the "perfectly scheduled" method in which there is some order in which nodes are allocated reserved time intervals for interface bus access.

[0005] Multi-access channels are being considered for interfacing the components of various satellite telephone systems currently under development. For example, multi-access channels are being considered for providing service for fixed user terminal (UT) systems planned for use in some satellite communication systems. A fixed UT, also called as a subscriber unit or communications terminal, can be a telephone, a wireless unit or a fixed station. The fixed UT is designed to operate from a fixed location. Generally, the fixed UT includes a radio antenna unit (RAU) and one or more telephone sets (also called desksets) connected to the RAU.

[0006] A remote location with party line service can be efficiently linked to a satellite or other wireless communications system through a radio antenna unit (RAU). An RAU is a transceiver, comprising well known elements, that transmits and receives a modulated carrier signal to and from the satellite communications system through an antenna. The RAU generally includes a receiver, a transmitter and other signal processing circuitry required for wireless operations. The RAU provides receive and transmit capabilities for one or more telephone sets. During transmission, the RAU accepts audio signals from the multiple phones. An audio coder-decoder (or audio codec) in the RAU digitizes the audio signals, which are then used to modulate the carrier signal that is radiated to a satellite (or other relay apparatus) by the antenna. During reception, the RAU receives an input signal comprising a modulated carrier signal from a satellite. The RAU demodulates the input signal to retrieve the digital audio signal. After which, the audio codec converts the digital audio signal to an analog audio signal, and causes the analog audio signal to be sent to the multiple phones. The RAU can use a variety of additional signal processing and control elements as desired, and known in the art.

[0007] There are various advantages associated with the fixed UT. The fixed UT is relatively inexpensive to implement. The fixed UT appears to be especially suitable at remote locations, such as oil rigs, isolated rural areas, and the like where it may not be economically feasible to link such remote areas by land lines.

[0008] The fixed UT also appears to be suitable in areas where a typical wireless local loop may not be economically feasible. In a wireless local loop, multiple cellular telephones communicate via a base station (cell site or station). The base station is connected to a public switched telephone network (PSTN) through land lines. However, in some areas, there may not be enough wireless telephone users to justify the installation of a base station. There may also be a lack of sufficient public telephone

network infrastructure. For those areas, a fixed UT with a single RAU being serviced by a satellite system or fixed relay stations is a preferred alternative.

[0009] The practical implementation of the fixed UT, however, still presents numerous challenges. What is needed is a suitable interface bus (that is, the multi-access channel) by which the RAU can be connected to multiple desksets. There is also a need for a method that will enable the desksets to share the interface bus and to prevent a collision in the interface bus when a plurality of desksets transmit packets concurrently. There is also a need for a control interface protocol that will enable the desksets to efficiently communicate with the RAU.

SUMMARY OF THE INVENTION

[0010] The present invention is directed to a control interface protocol for a wireless communication system such as a satellite telephone system. The satellite telephone system comprises a radio antenna unit (RAU) and a plurality of desksets connected to the RAU through an interface bus. A control interface protocol governs communications between the desksets and the RAU over the interface bus.

[0011] In one aspect, the present invention relates to a packet transmission system. According to the present invention, the RAU and the desksets communicate with each other using packets. Commands and data are sent in packets from the RAU to one or more desksets or from a deskset to the RAU. In one embodiment, each packet includes a start of header (SOH) byte, an address byte (ADDR), a command (CMD) byte, an argument (ARG) and a block check character (BCC). A packet begins with the SOH byte. The ADDR byte comprises a source and a destination address of the packet. The destination address allows an intended recipient to correctly receive the packet, while the source address allows a recipient to ascertain the originating source of the packet. The ADDR byte also includes a sequence number (SQN), as described in detail below.

[0012] According to one embodiment, the ARG is one byte when the most significant bit (MSB) of the CMD byte is 1, and the ARG has a variable length when the MSB of the CMD byte is 0. The BCC character is a parity check of all characters from SOH to ARG. The BCC character is formed by bit-wise exclusive OR (XOR) of each character sent, excluding those forming the SOH byte. All packets from the desksets, except negative acknowledgment (NAK) packets, are acknowledged by the RAU.

- [0013] In one embodiment, the packets originating from the desksets include a source address between 1 and n, where n is an integer number that is equal to the total number of desksets connected to the RAU. A packet to a specific deskset includes a destination address, that is, the address of the deskset. A packet originating from the RAU to all desksets includes a first default address (for example, zero). A packet originating from the RAU to a deskset that does not have an address includes a second default address (for example, an integer number).
- [0014] According to one embodiment, the packets are protected by horizontal and vertical parity checks. The BCC character is used for the vertical parity check. The bit at the end of each packet is used for the horizontal parity check.
- [0015] In another aspect, the present invention relates to a method for handling error control for packets sent to the desksets (also referred to as "terminals") by the RAU (also referred to as the "common node"). Each packet having modulo-sequential sequence numbers. The method includes the steps of sending a packet from the common node to one of the terminals; and sending a negative acknowledgment (NAK) from that one terminal to the common node when an error or unexpected sequence number is detected in the packet. That NAK includes the sequence number of the last valid packet received. Any lost packets are re-sent from the common node to that terminal when an unexpected sequence number is detected and a reboot command is sent from the common node to that terminal when the number of missed packets exceeds a predetermined threshold or when a NAK is received at the common node from the terminal.

BRIEF DESCRIPTION OF THE FIGURES

- [0016] The present invention will be described with reference to the accompanying drawings. In the drawings, like reference numbers generally indicate identical, functionally similar, and/or structurally similar elements. The drawing in which an element first appears is indicated by the leftmost digit(s) in its reference number.
- [0017] FIG. 1 illustrates an exemplary environment in which the present invention operates;
- [0018] FIG. 2 is a system level diagram according to one embodiment of the present invention;

- [0019] FIG. 3 illustrates a fixed user terminal system according to one embodiment of the present invention;
- [0020] FIG. 4 illustrates a deskset according to one embodiment of the present invention;
- [0021] FIG. 5 illustrates a standard data packet;
- [0022] FIG. 6 illustrates a fixed argument command byte;
- [0023] FIG. 7 illustrates a variable argument command byte;
- [0024] FIG. 8 illustrates an address/sequence byte;
- [0025] FIG. 9 illustrates time-outs based on addresses;
- [0026] FIG. 10 is a flow diagram illustrating transmission of data packets from desksets; and
- [0027] FIG. 11 is a flow diagram that illustrates processing of data packets by a deskset.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

- [0028] An exemplary wireless communication system in which the present invention is useful is illustrated in FIG. 1. It is contemplated that this communication system uses CDMA type communication signals, but this is not required by the present invention. In a portion of a communication system 100 illustrated in FIG. 1, one base station 112, two satellites 116 and 118, and two associated gateways 120A and 120B are shown for effecting communication with a fixed remote user terminal (UT) 104. Typically, the base stations and satellites/gateways are components of separate communication systems, referred to as being terrestrial and satellite based, although this is not necessary. The total number of base stations, gateways, and satellites in such systems depends on desired system capacity and other factors well understood in the art. UT 104 comprises a wireless communication device such as, but not limited to, a cellular telephone, a data transceiver, or a paging or position determination receiver, and can be hand-held or vehicle mounted as desired.
- [0029] In FIG. 1, some possible signal paths are illustrated for communication being established between UT 104 and base station 112, or through satellites 116 and 118, with gateways 120A and 120B. The base station-user terminal communication links are illustrated by line 132. The satellite-user terminal communication links between satellites 116 and 118 and UT 104 are illustrated by lines 142 and 144. The gateway-satellite communication links between gateways 120A and 120B and satellites

116 and 118 are illustrated by lines 146, 148, 150, and 152. Gateways 120A and 120B, and base station 112, may be used as part of one or two-way communication systems or simply to transfer packets or data to UT 104.

[0030] FIG. 2 is a system level diagram of a fixed UT 104 according to one embodiment of the present invention. In UT 104, an interface bus 204 connects a plurality of desksets 128a-128n to RAU 124. In operation, desksets 128 communicate over interface bus 204.

[0031] FIG. 3 illustrates a fixed UT system 300 according to one embodiment. Fixed UT system 300 comprises a common node (RAU 124), a shielded cable 308, a junction box (or a building entry panel) 312, an electronic industries association (EIA) 568 unshielded twisted pair (UTP) cable 316 (also referred to as a first UTP 316), one or more RJ-45 telephone jacks 320a, b, . . . n, one or more four pair RJ-45 telephone cords (also referred to as a second UTP 324a, b, . . . n), and one or more terminals (desksets 328a, b, . . . n). It should be understood that components, such as EIA 568 cables or conductors, RJ-45 jacks, and building entry panels, can be easily substituted with alternative components. For example, EIA 568 UTP cable can be substituted with other unshielded cables that are well known in the industry. Likewise, RJ-45 telephone jacks can be substituted with other types of jacks that are widely available.

[0032] In one embodiment, fixed UT system 300 operates on 12 volt DC power. Fixed UT 300, however, may be configured to operate at a different voltage level or on AC power. RAU 124 can be installed on a suitable support surface or object such as, but not limited to, a roof or pole, as is known. Junction box 312 can be installed nearby in a sheltered, but still outdoor, environment. Desksets 328 can be located indoors to protect them from weather.

[0033] In one embodiment, shielded cable 308 (18 wires) connects RAU 124 to junction box 312. Shielded cable 308 comprises a deskset interface (8 wires), a data interface (8 wires) and a radio power interface (2 wires). The deskset interface in turn comprises a control interface (2 wires), an audio interface (2 wires) and a power interface (2 wires). It will become apparent to one skilled in the art that other variations of the deskset interface, the data interface, and the radio power interface can also be used. For example, other numbers of conductors may be employed in certain applications.

[0034] Second UTP 324 (8 wires) and RJ-45 telephone jacks 320 connect one or more desksets 328 to RAU 124. According to one embodiment, up to 8 desksets can be

connected to junction box 312. It should be understood, however, that the present invention can easily be configured to support a higher or lower number of desksets.

[0035] Tables I, II and III below list some of the components shown in FIG. 3 according to one embodiment. Other components can be substituted for the listed components without departing from the spirit and scope of the present invention. Specifically, Table I lists constituent parts of shielded cable 308. Table II lists constituent parts of the deskset interface listed in Table I. Table III lists constituent parts of UTP cable 324.

Table I
Shielded Cable

deskset interface	8 wires
data interface	8 wires
radio power interface	2 wires

Table II
Deskset Interface

control interface	2 wires
audio interface	4 wires
power interface	2 wires

Table III
UTP Cable

control interface	2 wires
audio interface	4 wires
power interface	2 wires

[0036] FIG. 4 shows a deskset 328 according to one embodiment of the present invention. Deskset 328 represents a user's primary interface with the satellite telecommunications system. Deskset 328 comprises a handset 404, a ringer/speaker 408, which is also typically located on the handset or in the cradle, a multi-line liquid crystal display (LCD) 412, here four line, a keypad 416, and light emitting diodes (LEDs) 420. LCD 412 and keypad 416 are used to access and control the features of the

satellite telecommunications system. LEDs 420 indicate telephone mode and status of deskset 328. It should be understood, however, that other types or styles and configurations of desksets can also be used in the present invention.

[0037] In one embodiment, deskset 328 includes a processing element. In one embodiment, the processing element includes a microcontroller and a 2 wire EIA-485 based interface to a 386 type microprocessor in RAU 124. The processing element takes commands and data from RAU 124 and lights the LEDs, outputs commands and data to the LCD and controls various audio components. Also, deskset 328 scans the keypad and reports key state changes to RAU 124. Deskset 328 also performs certain diagnostics and reports errors to RAU 124.

[0038] In operation, RAU 124 sends commands and data in packets to one or more desksets 328. Likewise, desksets 328 send commands and data in packets to RAU 124. Note that desksets 328 do not send packets to other desksets 328. More specifically, RAU 124 sends high level commands and LCD data to one or more desksets 328. Desksets 328 send button and error data and negative acknowledgment (NAK) packets to RAU 124.

[0039] Data is transmitted as a data packet. More specifically, data is sent in asynchronous serial characters. In one embodiment, the characters are comprised of 11 bits, namely a start bit, 8 data bits, a parity bit and a stop bit. In one embodiment, the parity bit is even and the bit rate is 38,400 baud.

[0040] FIG. 5 illustrates the structure of data packet 500 according to one embodiment of the present invention. Data packet 500 comprises a start of header (SOH) byte 504, an address (ADDR) byte 508, a command (CMD) byte 512, an argument (ARG) byte 516 and a block check character (BCC) 520. It will be apparent to a person skilled in the art that a data packet having an alternate structure can be used as well.

[0041] Referring back to FIG. 5, the structure of data packet 500 begins with SOH byte 504. In one embodiment, SOH byte 504 is an ASCII (0x01) character. Following SOH byte 504 is ADDR 508. ADDR 508 contains the source and destination address of data packet 500. The most significant bit of ADDR 508 is a source bit. In one embodiment, a sequence number (SQN) of data packet 500 is contained in bits 4, 5 and 6 of ADDR 508.

[0042] The third byte of data packet 500 is CMD byte 512. All commands are coded as one byte command characters followed by one or more bytes of argument (ARG). If the MSB of CMD byte 512 is 0, the ARG is one byte. If the MSB of CMD byte 512 is 1,

the number of bytes in the ARG is of a variable length. FIGS. 6 and 7 illustrate a fixed argument command byte and a variable argument command byte, respectively.

[0043] Referring back to FIG. 5, data packet 500 concludes with a BCC character 516. BCC character 516 is a longitudinal parity check of all characters from SOH byte 504 to the end of the argument not including SOH byte 504. BCC character 516 is formed by bit-wise exclusive or (XOR) of each character, excluding SOH byte 504.

[0044] In an alternate embodiment, BCC character 516 may be a cyclic redundancy check (CRC) of all characters from SOH byte 504 to the end of the argument, including or not including SOH byte 504.

[0045] Packets sent by desksets 328 are normally acknowledged by RAU 124 with an acknowledge (ACK) N packet, where N is a deskset 328 address number. An exception to this rule is that negative acknowledgment (NAK) packets from desksets 328 are not acknowledged by RAU 124. The ACK N packet is an ASCII DLE (data link escape) character followed by a numeric character. The numeric character is formed by an XOR operation of 0xBØ with ADDR 508 received from deskset 328.

[0046] The packets are transmitted and received in predetermined maximum time periods. In one embodiment, the maximum time period between successive characters in a packet is 800 microseconds. Also, in one embodiment, if a deskset 328 is in the midst of receiving a packet and there is a pause of 1 ms or more, deskset 328 detects an error, and, therefore, discards the packet as erroneous, or having errors. If RAU 124 does not receive an entire packet in a 10 ms time period, that is, RAU 124 receives only a partial packet in 10 ms, it discards the partially received packet as erroneous. This allows desksets 328 to rapidly regain framing or alignment with frame boundaries or timing in the midst of packet errors. It should, however, be understood that the present invention can be configured to have other transmission and reception times for packets.

[0047] FIG. 8 illustrates the structure of an address (ADDR) byte 800. ADDR 800 comprises a source bit 804, a sequence number (SQN) 808 and a source/destination address 812.

[0048] In order to simplify processing of packets, all packets are tagged with a source bit 804 that indicates whether the source is RAU 124 or a deskset 328. Source bit 804 of a packet that originates from RAU 124 is 0. Source bit 804 of a packet that originates from a deskset 328 is 1.

[0049] Packets originating from RAU 124 and directed to one or more desksets 328 include a 3 bit SQN 808. In a preferred embodiment, the SQN is modulo-sequential;

that is the SQN is incremented for each sent packet, from binary 000 to binary 111, and then to binary 000 and so on. Desksets 328 receive and process SQN 808 of all packets transmitted by RAU 124. In one embodiment, a packet from deskset 328 is tagged with a SQN 0.

[0050] Packets are tagged with either a source or a destination address depending on the packet's origin. Packets sent from a deskset 328 to RAU 124 are identified with a source address. According to one embodiment of the present invention, if there are n (n is an integer) number of desksets 328 connected to RAU 124, then a packet from deskset 328 is identified with a source address of between 1 and n . Packets sent from RAU 124 to a specific deskset are addressed with the deskset's address (between 1 and n). Packets from RAU 124 to all desksets 328 are addressed with a first default address, for example, zero. Those desksets that do not have an address yet or are currently arbitrating for an address are assigned a second default address, for example, $n+1$. It should, however, be understood that the addressing schemes described above are presented as an example only, and the present invention can be configured to operate using other addressing schemes. Those alternative schemes will be apparent to persons skilled in the relevant art.

[0051] As stated before, the present invention is directed to a communications protocol for a control interface for satellite telephone systems. However, the present invention may be used to provide a control interface for any type of radio telephone system where one or more desksets are connected to some type of a radio unit, such as a cellular or a PCS transceiver, a voice band point to point radio, or a packet data radio capable of receiving or transmitting digitized voice. Further, the present invention may also be used to control telephone sets connected to a box that would digitize and packetize voice data for transmission over computer networks such as the internet.

[0052] In the present invention, the control interface is shared by multiple desksets. A set of rules or protocol determine access to the control interface. The protocol used is based on carrier sense multiple access with collision detection (CSMA/CD). The protocol utilizes time-outs based on address numbers in granting access to the interface bus after an active period, a collision or in response to any request from RAU 124.

[0053] In one embodiment, the following protocol governs access to the interface bus. Each deskset 328 has a different pre-assigned time-out designated by ta_1 , ta_2 , ta_3 , ..., or ta_n , where n is the address of the deskset. The time-out is the amount of time deskset 328 waits before initiating a packet transfer. The time-outs according to one

embodiment of the present invention are listed in FIG. 9. It should be understood that other time-outs can also be assigned to the desksets.

[0054] Desksets 328 monitor the interface bus at all times. Before a deskset transmits a packet, it ensures that the interface bus has been idle for at least the time-out period t_a . The deskset then transmits SOH byte 504. If the transmitting deskset successfully transmits SOH byte 504, it assumes that the interface bus is idle and transmits the rest of the packet. No other deskset transmits during the transmission time, because the transmission time of a packet is less than the time-outs t_a for other desksets. If the interface bus is in contention, but a collision is not detected, the parity fails, and the packet is subsequently discarded. The recipient of the packet checks the parity of each byte. The recipient of the packet checks the BCC character to see if any bits were received in error. A bit received in error is a bit that is a complement of the bit transmitted.

[0055] If SOH byte 504 transmission is unsuccessful, a collision is detected. Each deskset 328 then waits for a period t_a before re-transmitting. Since each deskset's t_a is different, one deskset transmits before the other desksets or RAU 124. The other desksets 328 sense bus activity and wait. Since the idle time between characters in a packet is longer than the shortest t_a , the erroneous SOH byte will time out and not cause a packet error. If RAU 124 receives two or more SOH byte characters in a row, it discards the first SOH byte and treats the last SOH byte as a valid SOH byte. If RAU 124 receives an SOH byte followed by a second SOH byte, it does not have a valid packet. An SOH byte is supposed to be followed by an SQN. So, in this later case, RAU 124 discards the first SOH byte and examines whether or not the remaining data forms a valid packet.

[0056] In an alternate embodiment, neither deskset 328 nor RAU 124 checks for a collision during transmission of SOH byte 504. Collision is only detected by checking the parity bytes.

[0057] Packets of information are protected by both vertical and horizontal parity checks. Parity checks are needed to detect errors due to line noise. Also, parity checks are needed to detect incorrect wiring. Parity checks also detect errors caused by collisions when two or more desksets send SOH byte 504 characters at exactly the same time.

[0058] In the event of either parity failures or BCC errors, the receiving deskset or RAU discards the packet as erroneous. RAU 304 and deskset 328 each handle packet errors

differently. RAU 124 acknowledges error free packets from desksets 328, but discards packets with errors. In contrast, deskset 328 negatively acknowledges, that is, sends a NAK packet, packets with errors or partial packets that it receives.

[0059] Packets originating from desksets 328 are acknowledged in one of two ways. Normal packets are acknowledged with the ACK packet. NAK packets are acknowledged through the receipt of a valid RAU packet with a valid SQN number.

[0060] If deskset 328 does not receive an ACK packet within a predetermined time period, it re-sends the last packet. In one embodiment, if deskset 328 does not receive an ACK packet within 50 ms, it re-sends its last packet at time $50\text{ ms} + t_a$. Deskset 328 does not transmit any new packet until the current packet is acknowledged. Deskset 328 repeatedly sends the current packet every $50\text{ ms} + t_a$ until it is acknowledged.

[0061] FIG. 10 is a flow diagram that illustrates transmission of packets from deskset 328. In step 1004, the transmission process starts. In step 1008, deskset 328 transmits a packet. In step 1012, deskset 328 determines if an ACK has been received. If an ACK has been received, the process moves to step 1016, where the process is terminated. If an ACK has not been received, the process moves to step 1020, where deskset 328 determines if a predetermined amount or time, or a timer, has expired. If the preselected time or timer has not expired, the process returns to step 1012. If the time has expired, the process moves to step 1024, where deskset 328 re-sends the packet. Next, the process returns to step 1012.

[0062] Packets originating from RAU 124 include SQNs. Each deskset 328 keeps track of the last valid SQN received, whether or not the packet was addressed to that deskset 128. If deskset 328 receives a packet with errors, it sends a NAK packet. The NAK packet includes the SQN of the last valid data packet received from RAU 124.

[0063] Desksets 328 detect missing packets by checking SQNs. If the SQN of the most recent packet is one greater than the SQN of the previous packet, the most recent packet is valid and is processed normally. If the most recent packet's SQN is the same as the SQN of the previous packet, the most recent packet is assumed to be a repeat, and is ignored. If the SQN of the most recent packet is one less than the SQN of the previous packet, the most recent packet is also ignored. Any other SQN causes deskset 328 to send a NAK packet.

[0064] RAU 124, upon receiving a NAK packet, sends the last, or last few, packets. For example, if the last packet transmitted by RAU 124 was packet #2 and RAU 124 receives a NAK packet #1, RAU 124 knows that deskset 328 has not yet received

packet #2. So, RAU 124 resends packet #2. If RAU 124 last sent packet #3, and receives NAK packet #1, it resends packets #2 and #3. If RAU 124 sends packet # 3, and it gets a NAK packet #4, it can't be sure how many packets deskset 128 has missed, and so it resets deskset 328. In one embodiment, RAU 124 sends a reboot command to deskset 328. The reboot packet causes deskset 328 to reboot, thereby resetting its internal state to a known initial state.

[0065] If multiple desksets 328 receive erroneous packets from RAU 124, only one NAK packet is sent from a deskset 328. Desksets 328 monitor the interface bus for other deskset NAK packets. If another deskset 328 sends a NAK packet with equal or lower SQN, the deskset that has not yet sent a NAK packet need not, and will not. FIG. 11 is a flow diagram that illustrates processing of data packets by deskset 328 according to one embodiment of the present invention. Referring now to FIG. 11, the process begins in step 1104. In step 1108, deskset 328 receives characters (for example, SOH, ARG, BCC, etc) and the characters are queued in the deskset's memory. In step 1112, deskset 328 determines if the interface bus has been idle for 1 ms or more. If the interface has been idle for 1 ms or more, the process moves to step 1116. Otherwise, the process returns to step 1108.

[0066] In step 1116, deskset 328 determines if the queue in the deskset's memory is empty. If the queue is not empty, the process moves to step 1120, where deskset 328 determines if the first byte in the queue is a null (that is, zero). If the first byte is a null, the process moves to step 1124, where the first byte (that is, the null) is discarded. If the first byte is not a null, the process moves to step 1128, where deskset 328 determines if the first byte in the queue is equal to an ASCII data link escape (DLE) character (0x10 or 10 hex). If the first byte in the queue is equal to a DLE, the process moves to step 1132, where deskset 328 determines if the queue has 2 or more characters. If the queue has 2 or more characters, the process moves to step 1136, where deskset 328 determines if the second byte is the deskset's ADDR. If the second byte is not the ADDR, the process moves to step 1140, where bytes that have already been processed are removed from the queue. Next, the process returns to step 1116. If the second byte is equal to the deskset's ADDR, the process moves to step 1152, where deskset 328 stops waiting for an ACK packet and the process then returns to step 1140.

[0067] Referring back to step 1132, if the queue does not have 2 or more characters, the process moves to step 1148, where the characters in the queue are discarded, deskset 328 sends a NAK packet, and the process returns to step 1104. If the first byte in the

queue is not equal to DLE, the process moves to step 1144, where deskset 328 determines if the first byte in the queue is an SOH byte. If the first byte in the queue is not an SOH byte, the process proceeds to step 1148. If the first byte is an SOH byte, the process moves to step 1156, where deskset 328 determines if the queue has 5 or more bytes. If the queue has 5 or more bytes, the process moves to step 1160, where deskset 328 determines if the most significant bit of the CMD byte is 1. In other words, deskset 328 determines if there is a variable length packet in the queue. If the most significant bit of CMD byte is 1, then the process moves to step 1164, where it is determined if there are enough bytes in the variable length packet. In one embodiment, if in addition to 5 bits in the ARG, there are 4 bytes for overhead, then there are enough bytes, and the process moves to step 1168. Otherwise, the process returns to step 1148.

[0068] Referring back to step 1168, deskset 328 determines if the BCC character is valid. If the BCC character is invalid, the process returns to step 1148. If the BCC character is valid, the process moves to step 1172, where deskset 328 determines if the packet is from RAU 124. If the packet is not from RAU 124, the process moves to step 1174, where it is determined if a NAK packet has been received. If a NAK packet has not been received, the process moves to step 1176, where the packet is removed from the queue because the packet is not from RAU 124, but rather from a deskset, and the process then returns to step 1104. If a NAK packet has been received, the process moves to step 1178, where it is determined if the ARG byte of the packet is equal to the SQN of the last valid packet. If the ARG is not equal to the SQN of the last valid packet, the process returns to step 1176. Otherwise, the process moves to step 1180, where NAK packets that are queued to be sent out are cleared and the process then returns to step 1176.

[0069] Referring back to step 1172, if deskset 328 determines that the packet is from RAU 124, the process moves to step 1182, where deskset 328 determines if the SQN is equal to the last valid SQN +1. If the SQN is equal to the last valid SQN +1, the last valid SQN is incremented and the packet is processed. Then the process moves to step 1188, where NAK packets are cleared, and deskset 328 stops repeating NAK packets. If the SQN is not equal to the last valid SQN +1, the process moves to step 1186, where deskset 328 determines if the SQN is equal to the last valid SQN. If the SQN is equal to the last valid SQN, the process returns to step 1188. If the SQN is not equal to the last valid SQN, the process moves to step 1190, where deskset 328 determines if the SQN is equal to the last valid SQN-1. If the SQN is equal to the last valid SQN-1, the process

returns to step 1176. Otherwise, the process moves to step 1192, where the packets in the queue are discarded, NAK packet is sent, and the process then returns to step 1104.

[0070] While an embodiment of the present invention has been described above, it should be understood that it has been presented by way of example only, and not limitation. Thus, the breadth and scope of the present invention should not be limited by the above-described exemplary embodiment, but should be defined only in accordance with the following claims and their equivalents.

What we claim as the invention is: